Exe 1, One-message ZK proof. (10 points) Prove Claim 1 in Lecture 7: Assume that $\mathcal{L} \subseteq \{0,1\}^*$ has a *one-message* ZK proof (even computational), with standard completeness and soundness,[1] then $\mathcal{L} \in \mathrm{BPP}$.

Bonus*: prove the above for 2-message protocols.

exercise 1 The case where the single message is from V to P, is quite trivial. In that case the PPT V can serve as a PPT algorithm that decides whether $x \in \mathcal{L}$ or not. So assume that the single message is from P to V. We'll use the following notations:

- Denote the single message send by P by m.
- For all algorithm under discussion (P, V, S), when we need to refer to element in their view we'll use the dot-notation. For example $P(x).m$ means the message m sent by P (when working on x). Also $S(x).m$ is the (single) message m as produced by the simulator S (when running on x)
- Denote V as: $V(x, m)$, where x is the input and m is the single message sent by P. Hence the interaction between P and V is: $V(x, P(x).m)$ (same as $(P, V)(x)$).

It's tempting to claim that the following PTT algorithm decides whether $x \in \mathcal{L}$:

Algorithm 0.1 (TRY)**.**

input*: x*

- *Run simulator S on x*
- *Apply the protocol: $< P, V > (x)$. As a message from P use: $S(x).m$*
- *return V's decision.*
- *Or shortly the algorithm is: return $V(x, S(x).m)$*

One could claim that since S is a 'good' simulator, $S(x).m$ is 'very close' to the $P(x).m$, hence the completeness of that algorithm would follow from the completeness of (P, V). That would probably be true if we would assume that our protocol is PZK. Since it's only CZK we need to work more ...

[1] That is, the completeness is $\frac{2}{3}$ and soundness error is $\frac{1}{3}$.

Start with a simple claim, that will be used also in the Bonus solution:

Claim 0.2. *Assume the we have a protocol, (P, V) for the language $\mathcal{L}$. Assume A is a PPT such that:*

- *for every $x \in \mathcal{L}$*

$$\Big|\mathsf{Pr}[(P, V)(x) = 1] \;-\; \mathsf{Pr}[A(x) = 1]\Big| = neg(|x|) \tag{1}$$

  Where the probability is taken over the random coin tosses by the algorithms: P, V, A.

- *for every $x \notin \mathcal{L}$*

$$\mathsf{Pr}[A(x) = 1] \;\le \frac{1}{3} \tag{2}$$

Then $\mathcal{L} \in BPP$

Proof of Claim 0.2. From (1) we get that there exist $N \in \mathcal{N}$, such that for every $x \in \mathcal{L}, |x| \ge N$ we get:

$$\Big|\mathsf{Pr}[(P, V)(x) = 1] \;-\; \mathsf{Pr}[A(x) = 1]\Big| \;\le\; \frac{1}{100} \tag{3}$$

For that specific $N \in \mathcal{N}$, we know that there are finitely many $x \in \mathcal{L}$ such that $|x| \le N$. Denote those values as $x_1, x_2, \ldots, x_k$. The following is a decision BPP algorithm for the language $\mathcal{L}$:

Algorithm 0.3 (B)**.**

<u>input</u>*: x*

- *if ($|x| \le N$)*
- *if($\exists\, i \quad 1 \le i \le k \quad : \quad x = x_i$)*
- *return 1*
- *else*
- *return 0*
- *return $A(x)$*

Claim 0.4. *B has completeness at least: $\frac{2}{3} - \frac{1}{100}$*

Proof of Claim 0.4. Suppose $x \in \mathcal{L}$.
If $|x| \leq N$ then $B(x) = 1$ with probability 1. For $|x| > N$, using (3) we get:

$$\mathsf{Pr}[B(x) = 1] = \mathsf{Pr}[A(x)) = 1] \ \geq \ \mathsf{Pr}[(P, V)(x) = 1] - \frac{1}{100}$$

Now using the completeness of the protocol (V, P), we get:

$$\mathsf{Pr}[B(x) = 1] \geq \ \mathsf{Pr}[(P, V)(x) = 1] - \frac{1}{100} \ \geq \ \frac{2}{3} - \frac{1}{100}$$

□

Claim 0.5. *B has soundness error of at most:* $\frac{1}{3}$

Proof of Claim 0.5. Suppose $x \notin \mathcal{L}$.
If $|x| \leq N$ then $B(x) = 1$ with probability 0. For $|x| > N$, it follow immediately from the second assumption that:

$$\mathsf{Pr}[B(x) = 1] \ = \ \mathsf{Pr}[A(x) = 1] \ \leq \ \frac{1}{3} \tag{4}$$

□

Combining the previous 2 claims it follows that $\mathcal{L} \in BPP$ □

We know that S's view is computationally indistinguishable from (P, V)'s view. In particular, that would be true to only a part of the view. The part we're interested in, is the pair $(x, P(x).m)$. So formally:

$$\{(x, P(x).m\}_{x \in \mathcal{L}} \approx_c \{(x, S(x).m)\}_{x \in \mathcal{L}}$$

The precise meaning of that is:
For every PPT algorithm A, and for every $x \in \mathcal{L}$ we have:

$$\Big| \mathsf{Pr}[A(x, P(x).m) = 1] - \mathsf{Pr}[A(x, S(x).m) = 1] \Big| \ \leq \ neg(|x|)$$

Where the probabilities are taken on the random coins tosses by A, S, P.
Intuitively, that means that we cannot distinguish between $(x, P(x).m)$ and $(x, S(x).m)$ for large enough $x \in \mathcal{L}$.
Since the above is true for every PPT algorithm A, it must also be true for V. Hence we get:

$$\Big| \mathsf{Pr}[V(x, P(x).m) = 1] - \mathsf{Pr}[V(x, S(x).m) = 1] \Big| \ \leq \ neg(|x|)$$

But writing $\mathsf{Pr}[V(x, P(x).m) = 1]$ is the same as $\mathsf{Pr}[(P, V)(x) = 1]$. So we get:

$$\Big|\mathsf{Pr}[(P, V)(x) = 1] - \mathsf{Pr}[V(x, S(x).m) = 1]\Big| \ \leq \ neg(|x|)$$

Now in order to apply Claim 0.2 to the algorithm $A(x) = V(x, S(x).m)$ we just need to make sure that:

$$\mathsf{Pr}[A(x) = 1] \ \leq \frac{1}{3} \tag{5}$$

So consider the following cheater prover P^*, that defined as:
$P^*(x) = S(x).m$. So for that prover, the protocol (P^*, V) is the same as A. Hence from the soundness of the protocol (P, V) we get:

$$\mathsf{Pr}[A(x) = 1] \ = \ \mathsf{Pr}[(P^*, V)(x) = 1] \ \leq \frac{1}{3} \tag{6}$$

So from Claim 0.2 we get that $\mathcal{L} \in BPP$

Bonus

So assume we have a 2-message CZK, that decides $\mathcal{L}$, we'll show that $\mathcal{L} \in BPP$. If the first message is sent by P - it's the same as in previous answer. So assume V send a message to P, and then P return a message to V. We'll use the following notations/assumptions:

- m_1 will denote the first message (sent from V to P), m_2 the second.
- We'll treat V as a combination of 2 algorithms: (V_1, V_2). V_1 is responsible to produce m_1, V_2 produce the final result. more details:
- V_1's input is x - the common input to (P, V). V_1's output is a pair: (r, m_1) where r is the randomness used by V_1, m_1 is the message send to P. We'll write: $(r, m_1) \leftarrow V_1(x)$.
- V_2's inputs are (x, r, m_2). We assume that using the randomness r, V_2 can fully emulate V_1 work, and get to the point where V_1 sent m_1. m_2 is the message received from P. V_2's output is either 1 or 0 ($x \in \mathcal{L}$ or not). We'll write: $V_2(x, r, m_2)$
- For the prover P, its inputs are (x, m_1), its output is m_2. So we write: $m_2 \leftarrow P(x, m_1)$
- We'll use the dot-notation as before. So if X is any algorithm/protocol and i some item in its scope we'll write $X.i$ to refer it. So for example $(P, V)(x).m_2$ we mean the message m_2 during that interaction between P and V on x. And $S^*(x, m^*).m_2$ will refer to the message m_2 as simulated by S^* when it worked on inputs (x, m^*).

Let's look how the protocol (P, V) looks, in current notations:

Algorithm 0.6 (protocol: (P, V))**.**

input*: x*

1. $(r, m_1) \leftarrow V_1(x)$
2. $m_2 \leftarrow P(x, m_1)$
3. *return* $V_2(x, r, m_2)$

Consider the following cheater V^*. V^* get an auxiliary input: m^*. V^* act the same as V, but the m_1 it sends is always that auxiliary input it got. To put it formally, here is how the protocol (P, V^*) looks like:

Algorithm 0.7 (protocol: $(P, V*)$)**.**

input: *x, m^* to V^**

1. $(r, m_1) \leftarrow V_1(x)$
2. $m_2 \leftarrow P(x, m^*)$ *NOTE: m^* was sent to P not m_1*
3. *return* $V_2(x, r, m_2)$

Since it's is a CZK, we know that there exist a simulator $S^*(x, m^*)$, that simulates $(P, V^*)(x, m^*)$. Consider the following PPT A, that get x as input.

Algorithm 0.8 (A)**.**

input: *x*

1. $r, m_1 \leftarrow V_1(x)$
2. $m_2 \leftarrow S^*(x, m_1).m_2$
3. *return* $V_2(x, r, m_2)$

Here is the main claim:

Claim 0.9. *For every $x \in \mathcal{L}$ we have:*

$$\Big|\mathsf{Pr}[(P, V)(x) = 1] \ - \ \mathsf{Pr}[A(x) = 1]\Big| = neg(|x|)$$

Where the probability is taken over the random coin tosses by the algorithms: P, V, A.

Proof of Claim 0.9. Consider first the following 'intermediate' algorithm A':

Algorithm 0.10 (A')**.**

input: *x*

1. $r, m_1 \leftarrow V_1(x)$
2. $m_2 \leftarrow (P, V^*)(x, m_1).m_2$ *// we replaced S^* with (P, V^*)*
3. *return* $V_2(x, r, m_2)$

So A' differ from A only in the second line where it use the m_2 of (P, V^*) instead of m_2 of S^*. Since $view(S^*(x, m^*))$ is computationally indistinguishable from $view((P, V^*)(x, m^*))$ we conclude that:

$$\left| \mathsf{Pr}[A'(x) = 1] \ - \ \mathsf{Pr}[A(x) = 1] \right| = neg(|x|) \tag{7}$$

Now consider the algorithms A', and the protocol (P, V) (0.6). The only difference is line 2 when they calculate m_2.
$A'.m_2$ is the m_2 generated by $(P, V^*)(x, m_1)$. But since the auxiliary input to V^* is m_1, (the output of $V_1(x)$), we get that:
$A'.m_2 = P(x, m_1)$ - exactly the m_2 as in $(P, V)(x)$ (not just close, but exactly the same distribution). We conclude:

$$\mathsf{Pr}[A'(x) = 1] \ = \mathsf{Pr}[(P, V)(x) = 1] \tag{8}$$

Combining (7) and (8) we get the desired result:

$$\left| \mathsf{Pr}[(P, V)(x) = 1] \ - \ \mathsf{Pr}[A(x) = 1] \right| = neg(|x|) \tag{9}$$

□

Claim 0.11. *For every $x \notin \mathcal{L}$ we have:*

$$\mathsf{Pr}[A(x) = 1] \ \leq \ \frac{1}{3} \tag{10}$$

Proof of Claim 0.11. Consider the following cheater prover P^*:

Algorithm 0.12 (P^*)**.**

input*: x, m_1*

1. *return* $m_2 \leftarrow S^*(x, m_1).m_2$

Using that prover, the protocol (P^*, V) will be:

Algorithm 0.13 (P^*, V)**.**

input*: x*

1. $r, m_1 \leftarrow V_1(x)$
2. $m_2 \leftarrow S^*(x, m_1).m_2$
3. *return* $V_2(x, r, m_2)$

But this is exactly Algorithm 0.8 (algorithm A) !!! So from the soundness of the protocol (P, V) we know:

$$\mathsf{Pr}[A(x) = 1] \;=\; \mathsf{Pr}[(P^*, V)(x) = 1] \;\leq\; \frac{1}{3} \tag{11}$$

$\square$

Now applying Claim 0.2 we get that $\mathcal{L} \in BPP$

Some remarks:

- Note that for 1-message we just used the simulator for the honest verifier. Hence we actually proved a stronger result, that is:
  Every language $\mathcal{L}$ that has a 1-message honest-ZK proof is in BPP
- Could it be done also for 2-message ? Probably not, because that would yield that GNI is in BPP. Recall that the protocol we saw in class for GNI was a ZK protocol for the honest verifier ....
- It's interesting to consider what will go wrong in our "2-message ZK implies BPP" proof technic if we try to apply it to the probably false consequence "3-message ZK implies BPP". So if one tries to imitate that proof, then he also consider a cheater verifier V^* that output as (now the second) message its auxiliary input, and then taking a simulator S^* for that verifier. Denote the first message (now send by the proover) by m_0. Then, one could try to create the following PPT:

  Algorithm 0.14 (3-messages)**.**

  input*: x*

  1. *create m_0 with a good distribution.*
  2. *$r, m_1 \leftarrow V_1(x, m_0)$*
  3. *$m_2 \leftarrow S^*(x, m_1).m_2$*
  4. *return $V_2(x, r, m_2)$*

  The first problem arise is how do we create that m_0. But this can be achieved quite easily by using the simulator of the honest verifier for example. So what would fail ?
  The problem is that when we run the S^* in line 3, it won't be synchronize with the m_0 we produced at line 1. S^* would produce 3 messages $m_0^{S^*}, m_1^{S^*}, m_2^{S^*}$ with a good distribution, but the distribution of $m_0, m_1^{S^*}, m_2^{S^*}$, probably won't be good. Hence we can't guaranty that V_2 would accept all $x \in \mathcal{L}$ (with high probability)